scrapy框架梳理

scrapy五大组件和工作流程

```
【1】引擎Engine
【2】爬虫程序Spider
【3】调度器Scheduler
【4】下载器Downloader
【5】项目管道Pipeline

爬虫项目启动时,引擎找到爬虫文件所要第一批要抓取的URL地址,交给调度器入队列,调度器出队列,交给下载器下载,下载完成后将response交给爬虫文件做数据解析提取,提取出来的数据交给项目管道去处理,如果有需要继续跟进的URL地址,则再次交给调度器入队列,如此循环
```

Scrapy写爬虫项目流程

```
【1】创建项目和爬虫文件
   scrapy startproject 项目名
   cd 项目名
   scrapy genspider 爬虫文件名 允许抓取的域名
【2】定义要抓取的数据结构(items.py)
【3】爬虫文件解析提取数据(爬虫文件名.py)
【4】项目管道负责数据处理(pipelines.py)
【5】全局配置(settings.py)
【6】运行爬虫(run.py)  scrapy crawl 爬虫名
```

Scrapy项目的启动流程

```
【1】方式一: 通过start_urls变量启动
【2】方式二: 通过重写start_requests()
   def start_requests(self):
       生成所有要抓取的URL地址,交给调度器入队列
```

Scrapy多级页面数据抓取思路

```
class TencentSpider(scrapy.Spider):
    name = 'tencent'
    allowed_domains = ['tencent.com']
    start_urls = ['http://tencent.com/']
```

```
    def parse(self, response):
        """一级页面解析函数"""
        li_list = response.xpath('xxxx')
        for li in li_list:
            item = TencentItem()
            # 提取具体数据
            yield scrapy.Request(url=xxx, meta={'item':item}, callback=parse_two)

    def parse_two(self, response):
        """二级页面解析函数"""
        item = response.meta['item']

        yield item
```

settings.py中常用变量

```
【1】USER_AGENT = ''
【2】ROBOTSTXT_OBEY = Fasle
【3】CONCURRENT_REQUESTS = 32
【4】DOWNLOAD_DELAY = 1
【5】DEFAULT_REQUEST_HEADERS = {}
【6】COOKIES_ENABLED = False
    False: 找 DEFAULT_REQUEST_HEADERS = {'Cookie':''}
    True:  找爬虫文件中 yield scrapy.Request(url=xx, callback=xx, cookies={})
【7】ITEM_PIPELINES = {}
【8】FEED_EXPORT_ENCODING = 'utf-8'
```

Scrapy数据持久化梳理

```
【1】csv json
    scrapy crawl 爬虫名 -o xxx.csv
    scrapy crawl 爬虫名 -o xxx.json
【2】MySQL、MongoDB
    2.1》pipelines.py中新建管道
        from .settings import *
        class XxxPipeline(object):
            def open_spider(self, spider):
                pass

            def process_item(self, item, spider):
                return item

            def close_spider(self, spider):
                pass
    2.2》settings.py中开启管道
        ITEM_PIPELINES = {'项目名.pipelines.类名':优先级}
【3】redis
    利用scrapy_redis模块
```

Scrapy分布式爬虫梳理

```
【1】分布式爬虫原理
    多台主机共享一个爬取队列,利用redis中的集合实现
【2】具体实现(settings.py)
    2.1》先写正常的Scrapy爬虫项目
    2.2》配置settings.py为分布式
        重新指定调度器    : SCHEDULER = 'scrapy_redis.scheduler.Scheduler'
        重新指定去重机制 : DUPEFILTER_CLASS = 'scrapy_redis.dupefilter.RFPDupeFilter'
        不清除请求指纹    : SCHEDULER_PERSIST = True
        Redis主机地址    : REDIS_HOST = 'IP'
        Redis端口号      : REDIS_POST = 6379
        Redis的管道     : ITEM_PIPELINES = {'scrapy_redis.pipelines.RedisPipeline':200}
```